DATA WAREHOUSE

Business Intelligence I
Group 8

Francisco Correia 20221306
Joana Dray 20220149
Rodrigo Freire 20221292
Santiago Carvalho 20221027
Valeria Caraus 20220316

SOLUTION

BI

State of Iowa
ALCOHOLIC
BEVERAGES DIVISION

TABLE OF CONTENTS

BI

TABLE OF CONTENTS

BI

01. INTRODUCTION

Nowadays, companies operate in a VUCA environment – volatile, uncertain, complex, and ambiguous, in which the market is constantly changing. This new reality is also highly competitive, and every point of advantage is needed to make a difference in the consumers experience and in order to make smart business decisions. Thus, it is of absolute need that businesses gather the tools to analyse, respond and plan to improve their strategy in different fields. Here lies the importance of data, the capacity of processing/analysing it into quality information and the need to store effectively.

Business intelligence makes it possible to leverage software and services, transforming data into business insights that help in strategic and tactical decisions throughout all levels of an organization. Business data is now a part of the company's backbone, and it has become mandatory that we have the ability to collect and retrieve it from all processes of the company, internal or external.

Taking this into consideration, we will present a report about the **alcoholic beverages** business in the **State of Iowa** – United States of America. **The data will be treated like it belongs to a normal company of alcoholic beverages** (selling and distribution of product). As such, we will explain the business, its challenges, and the problems we intend to resolve, relating it to the needs and objectives of the firm.

Finally, a Business Intelligence solution will be created and explained in the framework of a fully functional Data Warehouse.

02.
THE COMPANY

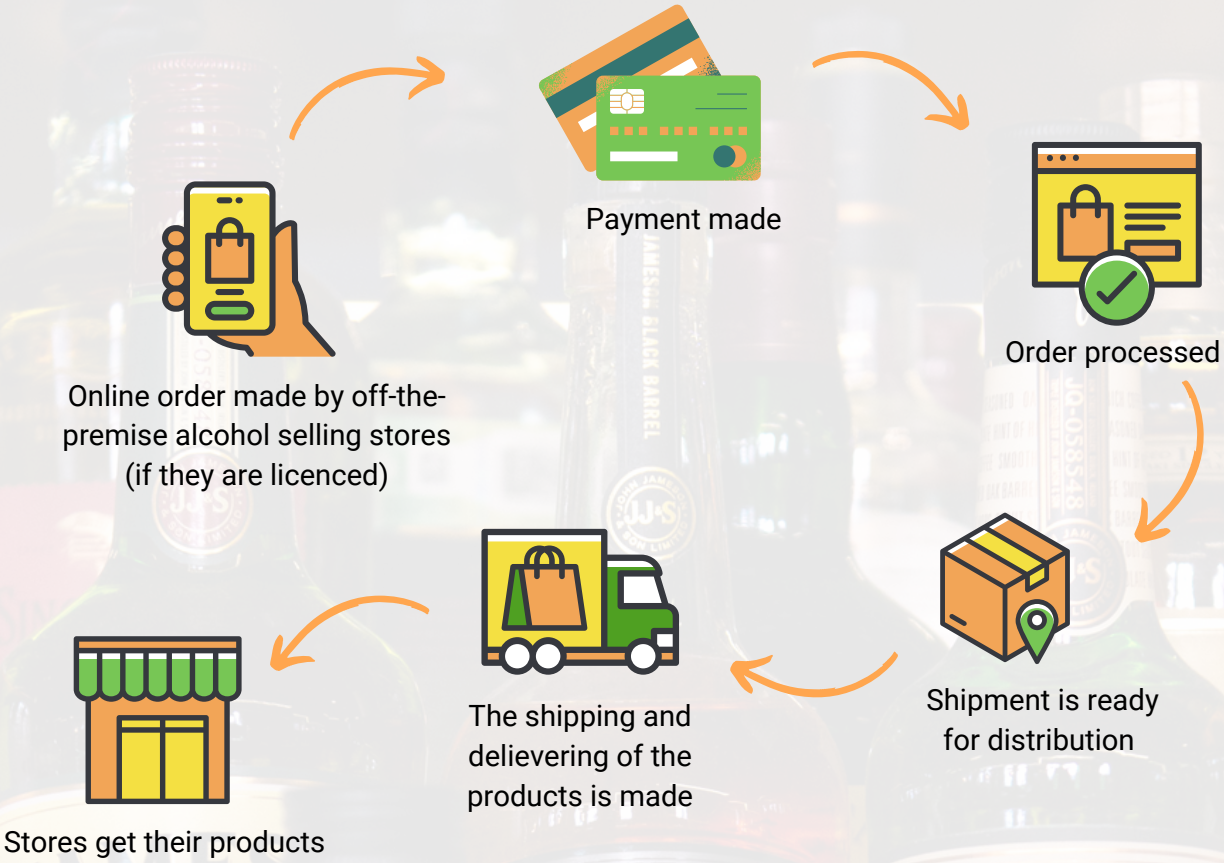
2.1. IOWA ALCOHOLIC BEVERAGES DIVISION

The **Iowa Alcoholic Beverages Division (ABD)** is based in the State of Iowa – United States of America and can be defined as an authority for controlling the alcoholic beverage market in the state. It is not only responsible for administrating and enforcing the alcoholic beverage laws in this state but has also **direct control over the distribution** at wholesale of alcoholic liquor to off-premise retailers, maintaining a monopoly in Iowa.

The organization **only** deals with the commercialization of **spirit drinks**, like vodka, gin, whisky, and so on. This means that beer and wine are **out** of the picture in this scenario. The Division facility is located in Ankeny and aims to serve the citizens of Iowa through responsible and efficient licensing, regulation, and distribution of alcohol.

The state of Iowa publishes monthly products and price listings, which contain a price booklet, as well as markups and markdowns. Due to legal reasons, every premise that sells alcohol in bottled form for off-premise consumption is required to hold a special liquor license, and only these selected stores are able to place an order in the online ordering system, which streamlines and simplifies the ordering process. Every alcoholic transaction made is then logged into the commerce system, which is what is going to be analyzed in this project.

ABD not only works as a **mass retailer** of spirit drinks in Iowa but it takes care of all the logistics, such as the **shipping** and **delivery** of the products.



Important note: since the original dataset is composed of more than 3 million rows of transactions, showing weekly liquor sales with multiple information, the group decided to only use a sample of all the information. Between 2018 and 2021, a certain number of monthly (random) transactions were chosen. We chose this format so that the data analysis was more similar to the original population, but it also means that the numbers we are going to analyze are not absolute.

According to the data available, the group was able to take conclusions about some broad aspects of Iowa Alcoholic Beverage Divisons's customers.

As seen in Figure 1, the **revenue** of the company has been demonstrating a pattern of positive **evolution**. Even though between the first three years (2018-2020) the total revenue was slightly the same, there was a notable jump to 2021, where the sales reached the biggest value, around $862000.

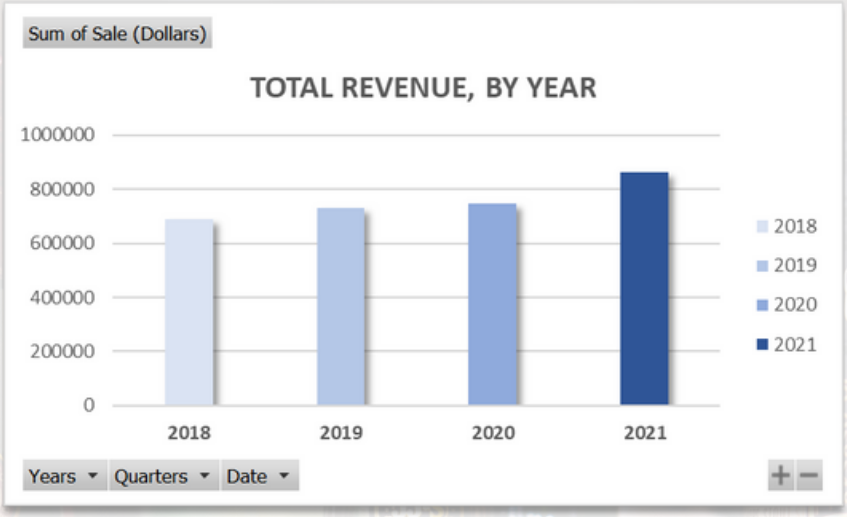


Figure 1 - Total Revenue by Year

Note: Figure 2 shows the Top 10 types of alcohol consumed in Iowa. The percentages are made out of these 10 categories, not from all the categories of the database.

Regarding the **most consumed types of alcohol**, it is quite remarkable that **American Vodkas** and **Canadian Whiskies** stand out from the rest of the categories. Therefore, it is noticed that the origin of each alcoholic beverage is a very important factor for the decision of consumers, since on one hand there are Canadian Whiskeys which take part of almost 20% of the sales of the top 10 categories, and on the other hand there are Tennessee Whiskeys with a very low percentage which goes around 5%.

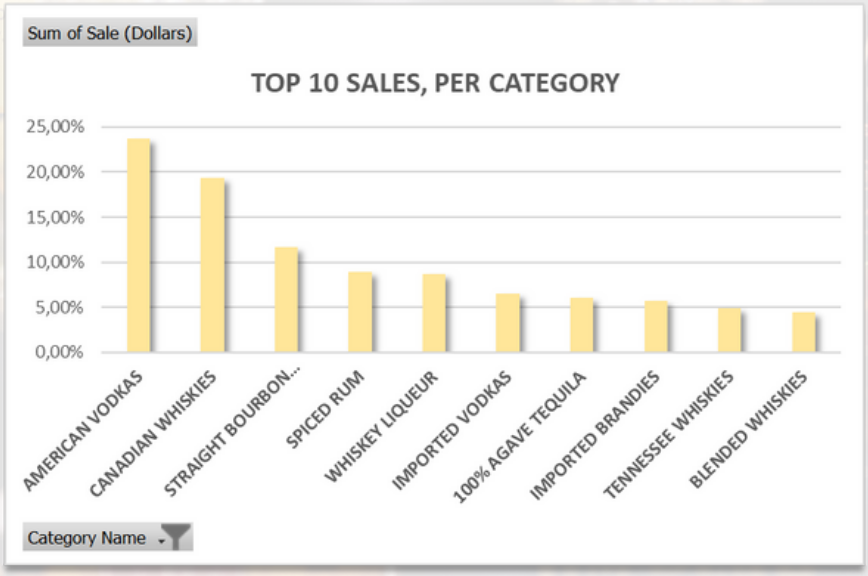


Figure 2 - TOP 10 Sales per Category

Since American Vodkas and Canadian Whiskeys are the main categories of alcohol sold, is expected that the products which are the most sold are inside of those categories. Therefore, the **product** that stands out the most compared with the rest is **Titos Handmade Vodka**, with a total sales of around $190000, followed by **Black Velvet** – a Canadian Whiskey.

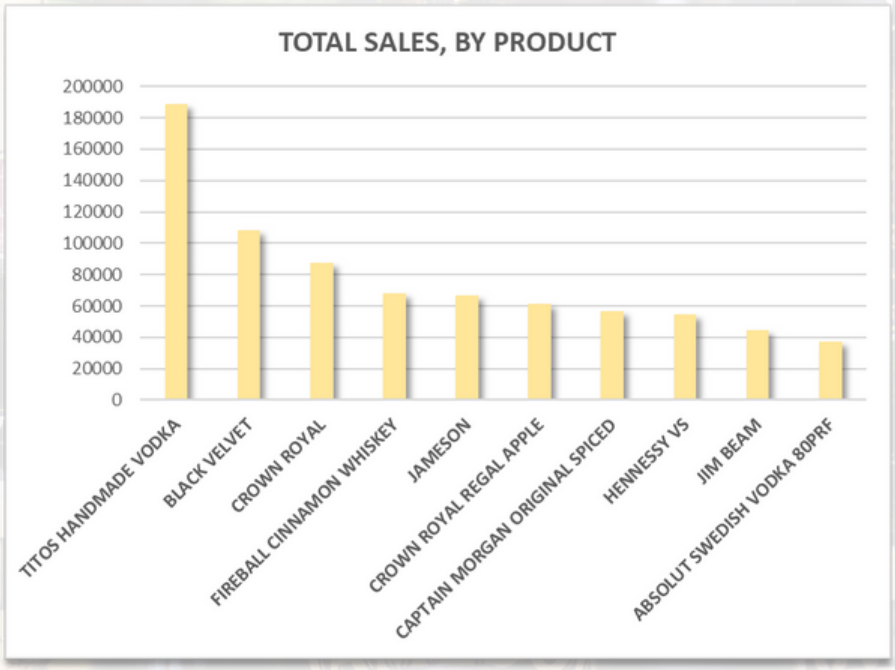


Figure 3 - Total Sales by Porduct

The **regions** are a very important factor to take into account when analyzing which cities consume more alcohol. As seen in Figure 4, **Des Moines** takes part in a very big amount of sales regarding our alcoholic beverages, standing out noticeably from all the other cities in the State of Iowa. This fact is interesting to understand the patterns of alcohol consumption according to each type of consumer.

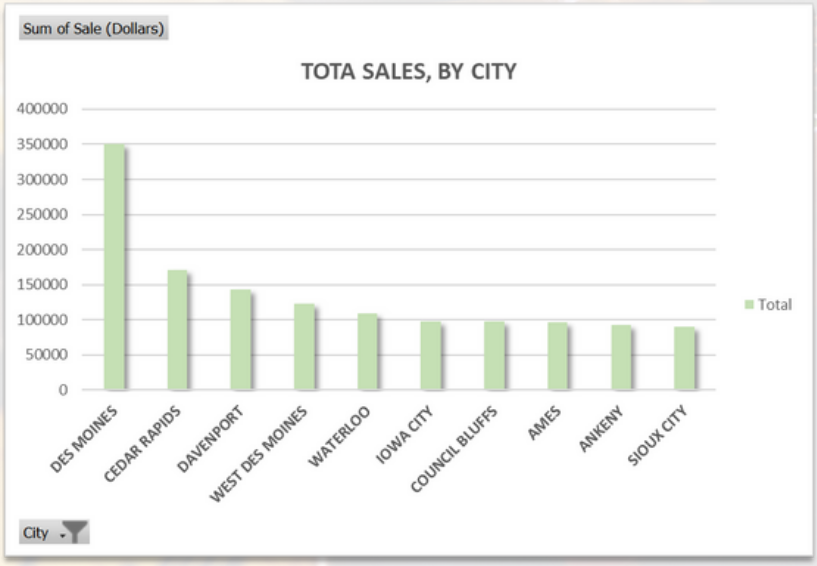


Figure 4 - Total Sales by City

Apart from the number of sales, from this database, it is possible to also analyze the **total amount of transactions** each city has done. As expected, **Des Moines** also takes the majority of the percentage of all the transactions done.

However, **Cedar Rapids** has a bigger presence here rather than in the number of sales because of the type of alcohol consumers prefer in each city, which depends on several factors, such as their prices and quality.

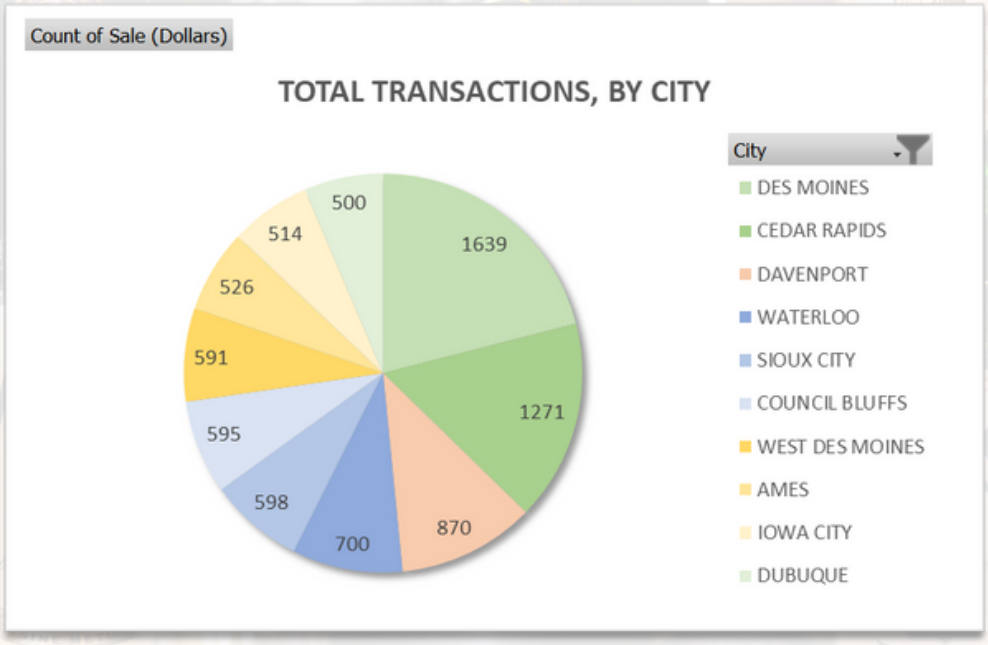


Figure 5 - Total Transactions by City

2.2. PROBLEM SCENARIO

This report analyses a representative sample of the transaction activity of alcohol in the United States, more specifically in the state of Iowa. This sample will be useful to answer several questions, such as: Which cities sell the most alcohol? Which is the most popular brand? How much is sold? Which type of alcohol is more consumed? Among others.

2.3. BUSINESS NEEDS

1. **Know which products have more impact on our company business.** It's essential to analyze the top products sold and understand which ones are more profitable. It's very relevant to understand if there's a correlation between the top most sold products and the most profitable ones. Analyzing it per year allows us to discover which products are more requested and which ones are maintained as most wanted/profitable and ones that fell on the market, potentially caused by Covid-19.

2. **Find out which products are "solid" in the market and the ones that contribute more to the company earnings.** We can also explore which category and sub-category of products is more demanding, depending on the season and in which county.

3. **Analyze the number of stores per county**, in order to know the county with more store frequency. With this information, our company can manage and plan strategies that allow us to work more efficiently.

4. **Discover who is our top monthly client in each city, per category/sub-category**. This can help us in different ways, like understanding what we are doing well with this client so that we can improve with others. Also, we can propose partnerships and consequently guarantee that we provide the best service to our best clients and maintain a good relationship with them.

5. **Know who is the shipper that we use the most to provide our service every month**. With this knowledge, we can propose and try different deals and partnerships depending on the company's strategy.

6. **Finally, we want to know which products are less profitable**, so that we can analyze if it makes sense to keep selling them. Also, it would be relevant to know which products are ordered the least and analyze their profit, so that we can see if they are worth keeping on the company's portfolio.

03. DATA SOURCES

Our main source database, provided by the Iowa Department of Commerce, Alcoholic Beverages Division, is in an Excel (and CSV) format and it contains the purchase information of Iowa Class “E” liquor licensees. Besides that, we have an auxiliary Excel file that provides information regarding the shipping service associated with the purchase order.

This Department has transactional records for many years but in this project, we will only analyze the purchase records from 2018 to 2021, which have 20 160 records.

3.1. DATA SOURCES DESCRIPTION

The main source database "IowaSource" contains four excel sheets with tables storing different information:

Purchase Orders – This table provides information regarding the orders placed to the Alcoholic Beverages Division. It contains the date of the purchase order and information regarding the sale, such as the quantity and volume sold, and total sale associated to each order. Also, it shows the ID of the product bought, the ID of the store who placed the order, the ID of the vendor who supplied that product to the Alcoholic Beverages Division and the shipping ID associated with the order.

Product – Provides detailed information about the products commercialized by the Alcoholic Beverages division, such as the product description, category, sub-category, pack, retail price and state cost (how much the division paid to the vendor).

Vendor – This table contains information about the vendors who commercialize the products to the Alcoholic Beverages Division and the contact details of the manager responsible for this process.

Store – Information about the stores that place orders, such as the identification and where it is located.

However, **not all attributes are relevant** to our analysis. Therefore, the selected columns that provide the necessary information regarding the purchase order are the following:

TABLE	RELEVANT FIELDS
PURCHASE ORDERS	Date; Bottles Sold; Sale (Dollars); Store ID; Product ID; Vendor ID; Shipping ID
PRODUCT	Product ID; Category Code; Category Name; Sub-Category Name, Item Description; Pack; Bottle Volume (ml); State Bottle Cost; State Bottle Retail
VENDOR	Vendor ID; Vendor Name; Manager Name; Contact Details
STORE	Store ID; Store Name; Address; City; Zip Code; County Number, County

Table 1 - Relevant main data source fields

From the auxiliary Excel file that contains the shipping details "IowaShipping", we consider that all of the fields are relevant.

Shipping – Information about the shipping and the carrier responsible. The company offers three types of shipping: Exclusive, Premium and Standard. Each type can choose among the following classes:

- 1st Class = up to 2 days delivery
- 2nd Class = 3 to 5 days delivery
- Standard Class = 5 to 10 days delivery

TABLE	RELEVANT FIELDS
SHIPPING	Shipping ID; Shipping Type; Shipping Class; Carrier Code; Carrier Name

Table 2 - Shipping fields

04.
DATA WAREHOUSE

A Data Warehouse is a central repository where consolidated data from multiple sources are stored. In other words, it's a structured and non-volatile single source of truth of a company.

The purpose of data warehouses is to gain valuable insights from data and consequently improve decision-making.

According to Bill Inmon, who is considered the father of the data warehouse, there are four unique characteristics which define a data warehouse. They are:

- **Subject-oriented:** Information in the data warehouse revolves around some subject, therefore it does not contain all company data but only the subject of interest.
- **Integrated:** The data within a data warehouse is usually derived from a wide range of sources so data might not be in the same format. So, a data warehouse is integrated when all the data from the different sources is consistent.
- **Time-variant:** A data warehouse contains a large amount of historical data which helps decision-makers to get valuable insights and predict trends.
- **Non-volatile:** Data flows into the data warehouse as is, so once it's there it cannot be changed or deleted.

The group decided to employ the Kimball's dimensional modeling approach, which is a data mart with a bottom-up procedure since this project has a limited scope.

Therefore, to design the data warehouse model, we will use the simplest style of a data mart, the **star schema**, which improves performance and has lower query times.

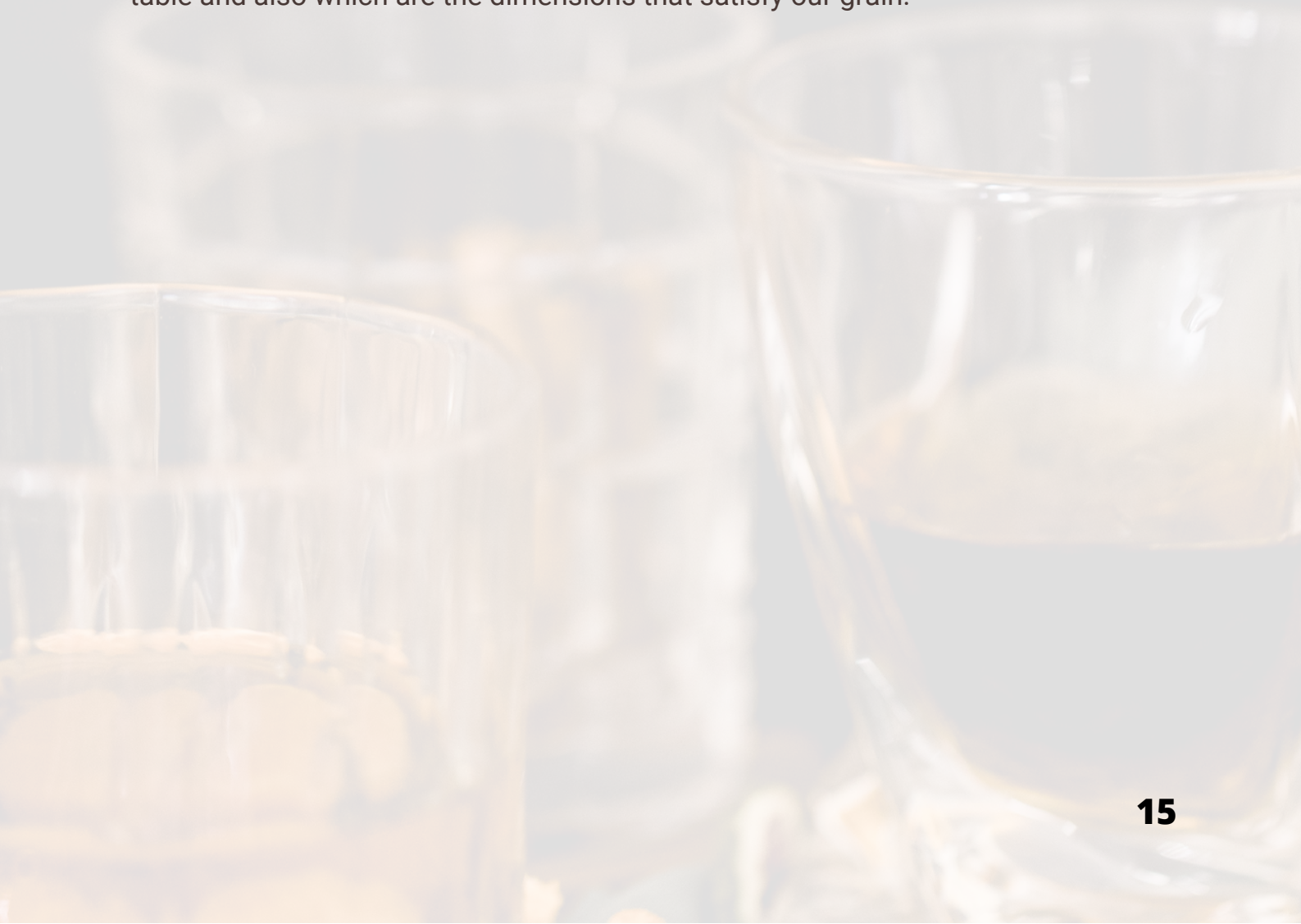
The **dimensional modeling** followed the steps below:

1. Identification of the business process
2. Identification of the grain
3. Identification of the dimensions
4. Identification of the facts

As introduced before, the business process that the data warehouse will represent is the **sales** generated from the purchase orders.

Regarding the grain, **each row represents an individual product purchased by a store from a vendor, and transported by a shipper, on a specific day.**

The business questions mentioned before help identify the measures for the fact table and also which are the dimensions that satisfy our grain.



4.1. MEASURES

In chapter 2, the business of the Alcoholic Beverages Division was described, and the respective business needs were identified. In order to select the measures, those business needs were translated in business questions, as shown in the table below:

BUSINESS QUESTION	MEASURES
1.What are the top 10 products sold (by sales quantity), per year?	Sales quantity
2. What are the top 10 products with higher profit, per month, season, and year?	Profit amount
3. What categories & sub-categories generated more profit along the years, per county?	Profit amount
4. Which months have the higher sales, per category, per year?	Sales amount
5. Which store has the highest order values, per city, per month?	Sales amount
6. Who is our top vendor, per category, per semester and year?	Cost amount
9. Which are the 20 products with less orders, per year?	Sales quantity
10. Are the top 20 products with less orders profitable, per year?	Profit amount

Table 3 - Business Questions and Measures

MEASURES	FIELD	DESCRIPTION
Sales Quantity	Bottles Sold	-
Sales Amount	Sale (Dollars)	-
Cost Amount	Pack; State Bottle Cost	State Bottle Cost x Pack
Profit Amount	Pack, State Bottle Cost, Sale (Dollars)	[Sale (Dollars)] – [Pack x State Bottle Cost]

Table 4 - Description of the measures and fields that provide them

4.2. DIMENSION TABLES

As mentioned before, dimensions are associated with the fact table. Each dimension table provides descriptive attributes of the fact table.

- **Dim_Vendor Table**

The vendor dimension stores information about the company from which the liquor was ordered.

COLUMN	DATA TYPE	DESCRIPTION
SK_Vendor	Int	Surrogate Key
BK_Vendor	Int	Business Key
Vendor_Name	Nvarchar(100)	Vendor's name that identifies the company brand from where the liquor was ordered
Manager_Name	Nvarchar(100)	Name of the manager of the brand where the liquor was ordered from
Contact_Details	Nvarchar(50)	Phone number of the manager
SCD_Start_Date	Datetime	Start date (record valid from this date)
SCD_End_Date	Datetime	Date of change (record valid until this date)

Table 5 - Vendor Dimension

- **Dim_Product Table**

The product dimension stores information about the liquors ordered and sold by the company.

COLUMN	DATA TYPE	DESCRIPTION
SK_Product	Int	Surrogate Key
BK_Product	Int	Business Key
Item_Description	Nvarchar(100)	Description of the individual liquor product ordered
Category_Code	Int	Category code of the liquor ordered
Category_Name	Nvarchar(100)	Name of the category of the liquor ordered
Sub_Category_Name	Nvarchar(100)	Name of the sub-category of the liquor ordered
Pack	Int	Number of bottles in a case of liquor ordered
Bottle_Volume	Int	Volume of each liquor bottle ordered in ml
State_Bottle_Cost	Decimal (18,2)	The amount that ABD paid for each bottle of liquor ordered
State_Bottle_Retail	Decimal (18,2)	The amount the store paid for each bottle of liquor ordered
SCD_Start_Date	Datetime	Start date (record valid from this date)
SCD_End_Date	Datetime	Date of change (record valid until this date)

Table 6 - Product Dimension

- **Dim_Store Table**

This table contains the identification of each store and information about its geographic location.

COLUMN	DATA TYPE	DESCRIPTION
SK_Store	Int	Surrogate Key
BK_Store	Int	Business Key
Store_Name	Nvarchar(100)	Name of the store who ordered the liquor
Address	Nvarchar(100)	Address of the store who ordered the liquor
City	Nvarchar(100)	City where the store that ordered the liquor is located
Zip_Code	Int	Zip code of the store that ordered the liquor is located
County_Number	Int	Iowa county number of the county where the store that ordered the liquor is located
County	Nvarchar(100)	County where the store who ordered the liquor is located
SCD_Start_Date	Datetime	Start date (record valid from this date)
SCD_End_Date	Datetime	Date of change (record valid until this date)

Table 7 - Store Dimension

- **Dim_Date Table**

This table contains values of dates with the following granularity: day, month, trimester, semester, and year.

COLUMN	DATA TYPE	DESCRIPTION
SK_Date	Int	Surrogate Key
Proper_Date	Date	Date of purchase - 20/12/2021
Full_Date	Nvarchar(50)	20 December 2021
Day_Number	Int	20
Day_Name	Nvarchar(50)	Monday
Month_Number	Int	12
Month_Name	Nvarchar(50)	December
Trimester_Number	Int	4
Trimester_Name	Nvarchar(50)	Trimester 4
Semester_Number	Int	2
Semester_Name	Nvarchar(50)	Semester 2
Year	Int	2021

Table 8 - Date Dimension

- **Dim_Shipping Table**

The shipping dimension stores information about the carriers of the products and the shipping, such as shipping type and shipping class.

COLUMN	DATA TYPE	DESCRIPTION
SK_Shipping	Int	Surrogate Key
BK_Shipping	Int	Business Key
Carrier_Code	Int	Number that identifies the carrier company
Carrier_Name	Nvarchar(100)	Name of the carrier company
Shipping_Category	Nvarchar(100)	Shipping category name
Shipping_Class	Nvarchar(50)	Shipping class name
SCD_Start_Date	Datetime	Start date (record valid from this date)
SCD_End_Date	Datetime	Date of change (record valid until this date)

Table 9 - Shipping Dimension

4.3. FACT TABLE

- **Fact_Sales Table**

This table shows information regarding the sales and profit of our business. It indicates when the purchase done and by which store, which vendor supplied the product to the Alcoholic Beverages Division, what product and category was purchased, who was the shipper and what was the shipping category and class.

COLUMN	DATA TYPE	DESCRIPTION
FK_Date	Int	Foreign Key
FK_Vendor	Int	Foreign Key
FK_Product	Int	Foreign Key
FK_Store	Int	Foreign Key
FK_Shipping	Int	Foreign Key
Sales_Quantity	Int	Number of bottles of liquor ordered by the store
Sales_Amount	Decimal (18,2)	Total sales of liquor order (number of bottles multiplied by the state bottle retail – amount the store paid for the purchase)
Cost_Amount	Decimal (18,2)	Total amount that the Alcoholic Beverages Division paid for each bottle of liquor (Number of bottles multiplied by the state bottle cost)
Profit_Amount	Decimal (18,2)	Profit amount

Table 10 - Sales Fact Table

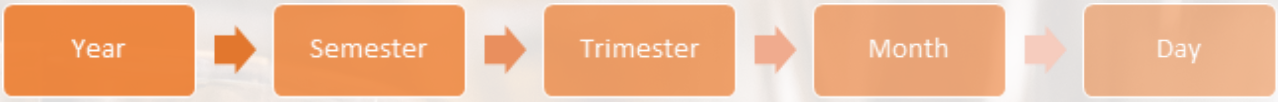
4.4. HIERARCHIES

A **hierarchy** is a set of levels that represent relationships between different attributes within a hierarchy and have many-to-one relationships between each other.

Product Dimension

The **product dimension** has three levels of depth. Each item exists in exactly one sub-category, but a sub-category may have many products. The same happens with the category – a category has many sub-categories, but a sub-category only has one category, hence the relationship is “many-to-one.” The level Category and Sub-Category have as attributes the name and code, the level Item has the ID, description, bottle volume and pack.

Date Dimension



The **date dimension** has five levels of depth, with the year as the top level and day as the lowest level of detail. All levels except the year have as attribute the number and name.

Store Dimension

The **store dimension** has three level of depth. The level county has as attributes the county number and name and the level city has the zip code.

Shipping Dimension

The **shipping hierarchy** has two levels, the type being the top level and class the bottom level.

Vendor Dimension

The **vendor** two levels of hierarchy, the company being the top level and the manager responsible for the sector at the bottom level. The attributes of the level Vendor are number and name, and the attributes for the level Manager are name and contact details.

4.5. SLOWLY-CHANGING DIMENSIONS

Dimension values may change over time, but how the change is handled depends on the value the management places on knowing the historical values of that dimension.

Type 0

- **Date Dimension -** In type 0, dimension attributes never change. For example, the date 25-12-2022 will always have the month 12 as December, the day will always be Sunday and it will belong to the 2nd semester of the year 2022.

Type 2

The following dimensions contain relevant information for our analysis and have attributes that can change, so to deal with this we will employ the Type 2 method of Slowly-Changing Dimensions.

With this method, whenever it's necessary to change a value, a new record is added, and a new surrogate key is assigned to the record. In order to avoid confusion and know which is the most recent and active record, two new columns will be added: SCD_Start_Date and SCD_End_Date.

The column SCD_Start_Date refers to the date when the attribute value was updated, and SCD_End_Date refers to the last day before it was changed. In other words, it shows the dates in which that record was valid for. If a value has not suffered any change, the SCD_End_Date is set to NULL.

- **Product Dimension -** There is a possibility that some characteristics of our products may suffer changes in the future, such as the size of a bottle may change so consequently the volume that it can hold will also change. The prices of the products that were bought and sold may also fluctuate, depending on the market competitiveness and demand.

- **Store Dimension -** Supposing that a store is reassigned to a different city or county, attributes that provide detail on the geographic location will also change. Since it may be of our interest to see changes in sales according to the location of the store, it’s important to store this historical information.

- **Vendor Dimension -** The managers of the company and associated contact details may change, and the information must be updated accordingly. Even though the possibility is lower, our vendors might suffer changes regarding their code or name of the company (because management might change, company might have been acquired by another, among others).

- **Shipping Dimension -** Similarly to the case above, the carrier code and name might change.

4.3. STAR SCHEMA MODEL

The star schema is represented by a central fact table and several dimension tables connected to the fact table, usually with a relationship of “one to many” – the "one" side being the dimension-type table while the "many" side is always a fact-type table.

Even though this is a denormalized schema, it is easier to understand and simple to implement. Also, accessing data is faster since there's no need to join various tables when querying the data to generate results.

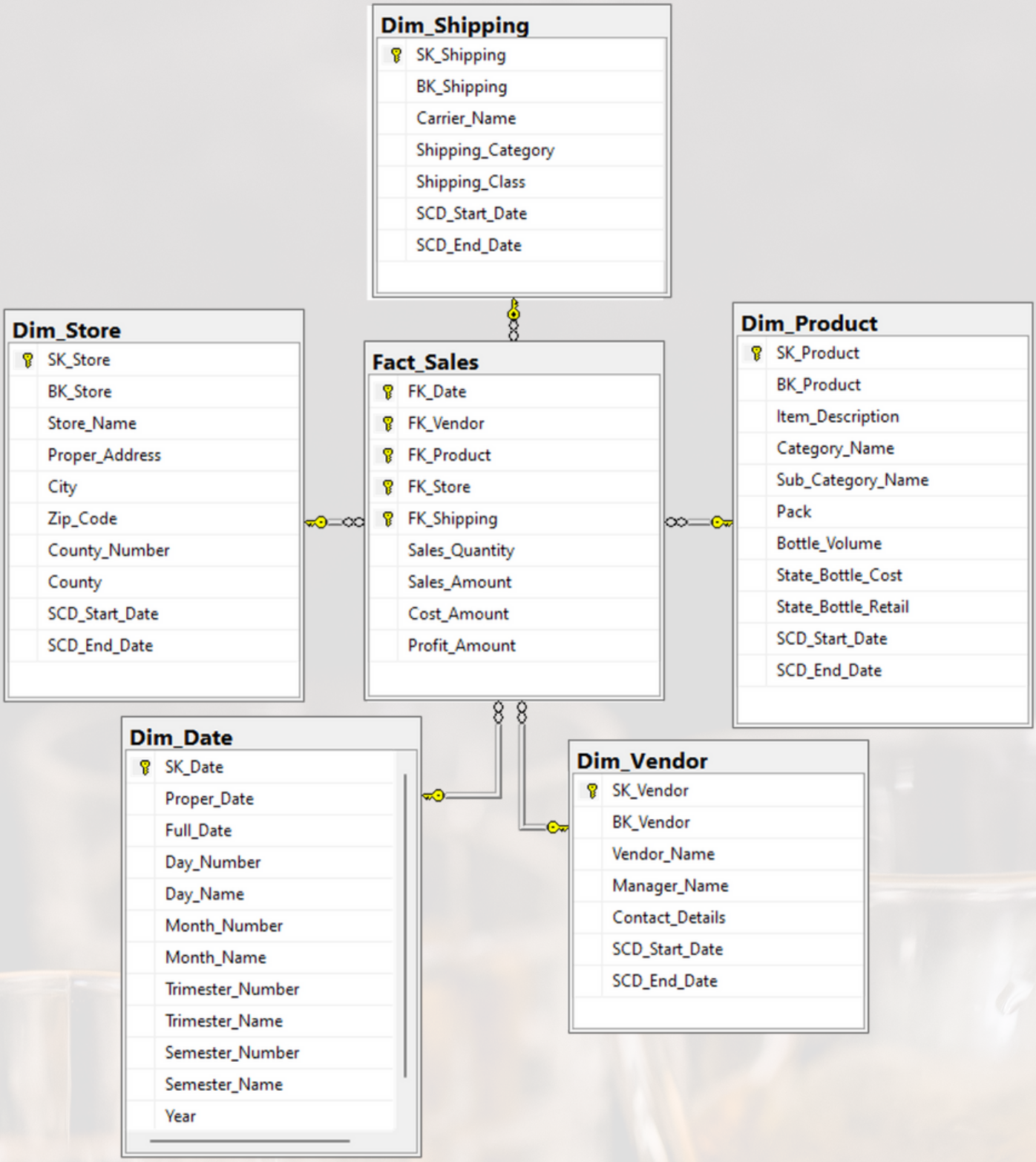


Table 11 - Star Schema

05.
ETL PROCESS

After designing the Data Warehouse, the following step stands for describing the ETL process, which has three important phases. The first step of this process is **Extraction**, where the organization´s data is selected from the original data source. Afterward, comes the **Transformation** phase, where the data is converted and transformed in accordance with the form the Data Warehouse previously designed needs. This stage sometimes requires a cleansing of the data in order to remove eventual errors that might exist. Finally, the third and final step of the ETL process is the **Loading** of the data that was transformed into the Data Warehouse.

The ETL process was based on the following **sources**:

- An Excel file with all the organization´s data (Iowa_DB.xlsx) was used on SQL. These files include information regarding the stores, vendors, products, sales, etc.
- An Excel file with date information.
- An Excel file with shipping information.

Between the data Extraction and the following steps, there is often a step called **Staging Area**. This process is responsible for transforming the data – by transforming, cleansing, validating, and/or merging it – as it is ready for being loaded into the Data Warehouse. Therefore, this is an intermediate storage area where the data is quickly processed from its data sources, minimizing the impact of the sources.

This whole process will consist of three different **packages**: first, the extracted data will be loaded into the Staging Area; afterward, this data will be loaded into the Data Warehouse; and finally, it comes to the execution of a Master Package, that executes both the Staging Area and Data Warehouse.

5.1. STAGING AREA

In order to perform the data loading process of our dataset, the method chosen was the **Incremental Load**. This technique is faster and handles big datasets more easily than the Full Load since there is less amount of data to connect with.

Incremental Load works as a selective method of moving data from one system to another, where the selected data is the most recently created or updated. This method compares the incoming data from the source system with existing data present in the destination. Therefore, Incremental Load allows loading updated data into a destination, ignoring the data that is identical in both source and destination.

The ETL process in the staging area begins by setting up the connections to the data sources. This project required the original excel files to be converted into multiple Flat Files for the connections to work properly:

- Date Connection
- Vendor Connection
- Vendor Connection (Manager Info)
- Product Connection
- Purchase Orders Connection
- Shipping Connection
- Shipping Connection (Carrier Info)
- Store Connection (File 1)
- Store Connection (File 2)

The group first started by doing a Full Load, and made the required changes along the way to have an Incremental Load as the final output.

5.1.1. STAGING AREA - CONTROL FLOW

The first task in this package is the **Log Start of ETL,** which registers the start of the ETL process in the table Log_Stg_Etl. For this SQL Task, it was necessary to create a new variable:

Name	Scope	Data type	Value	Expression
ETL_NAME	Package	String	ETL ID is: 07/01/2023 19:15:28	"ETL ID is: " + (DT_WSTR, 30)@[System::StartTime]

Figure 6 - Variable Creation (ETL_NAME)

In this way, it's possible to capture the beginning and ending timestamps for the ETL process, by keeping a record of the time the process started and ended.

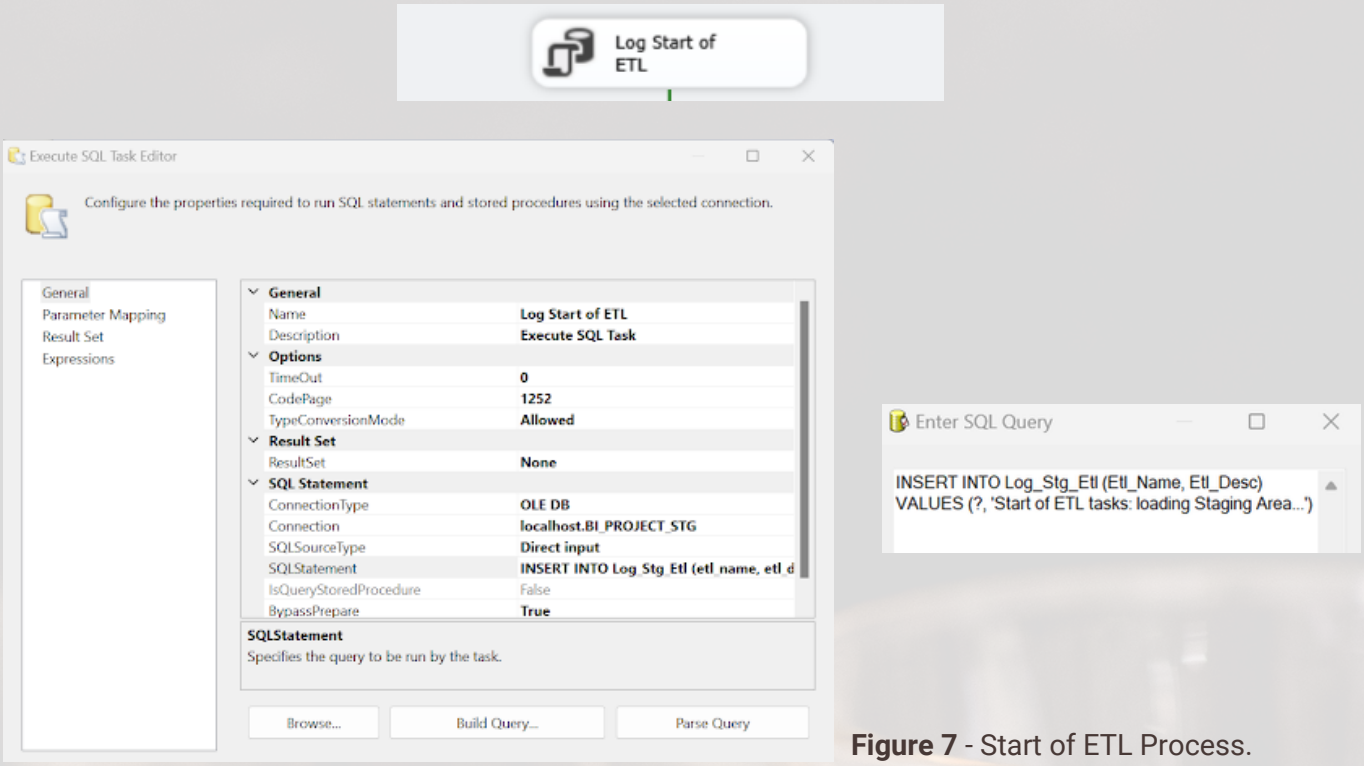
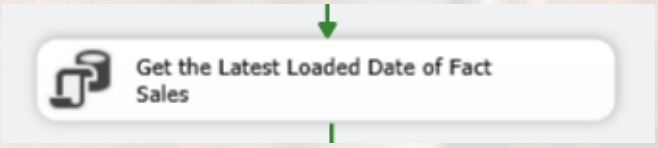


Figure 7 - Start of ETL Process.

In order to perform the Incremental Load of the Staging Area, it was necessary to create the variable Latest_Sales_Loaded_Date, which will store only the latest date values of the Fact Table. This variable will allow us to the extraction of the most recent facts which are already loaded in order to load only the ones that occurred after those dates.



Name	Scope	Data type	Value	Expression
ETL_NAME	Package	String	ETL ID is: 07/01/2023 19:15:28	"ETL ID is: " + (DT_WSTR, 30)@[System::StartTime]
Latest_Sales_Loaded_Date	Package	DateTime	22/12/2022 15:39	

Figure 8 - Variable Creation (Latest_Sales_Loaded_Date).

Afterward, we created a **container Delete Dim** and **Fact** with the purpose of eliminating the data from all the tables of the Staging Area. This way, we were able to delete and truncate the dimensions and the fact table, respectively.

This process was possible to do since there were no relationships between any of the tables required for the Staging Area that could raise errors, so we were able to clean all the tables simultaneously.

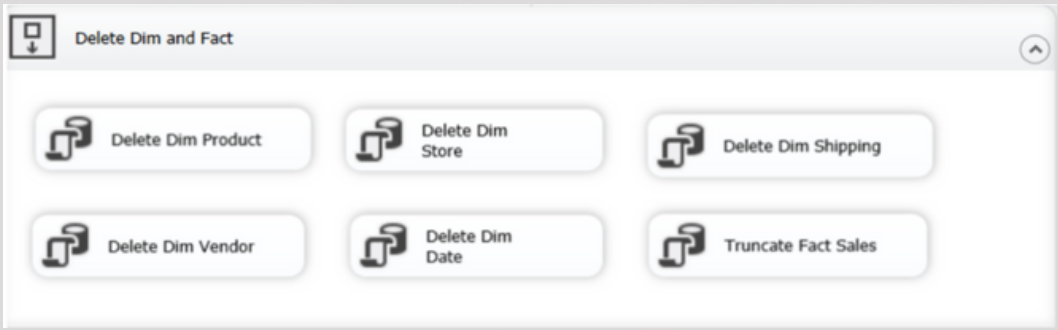
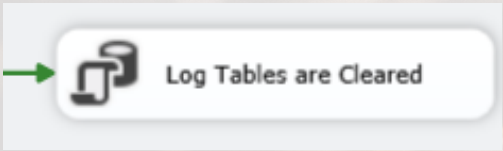


Figure 9 - Deletion of Dimensions and Fact Table.

We proceeded by creating a data flow task called **Log Tables are Cleared,** which enables the system to register the deletion and truncation process.



After completing these tasks, the next step was loading the dimensions and the fact tables into the Staging Area. As mentioned before, at this stage the tables still have no relationship with each other, so it was possible to load all tables in the staging area at the same time.

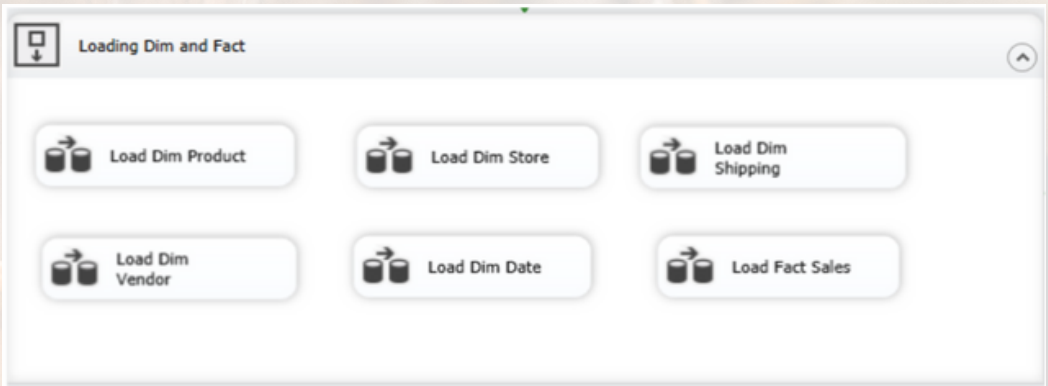
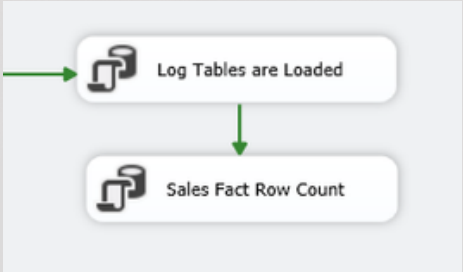


Figure 10 - Loading of Dimensions and Fact Table.

Next, the data flow task **Log Tables are Loaded** was created, with the purpose of registering the loading of the tables.
Also, in order to register the number of rows loaded for the fact table, we proceeded with the creation of the data flow task **Sales Fact Row Count**.



5.1.2. STAGING AREA - DATA FLOW

Load dimension tables

- **Load Dim Product**

The Product Dimension requires data from the Product Connection, which contains all the information regarding our products. We used the **Multicast** to divide the inputs into two outputs:

- Load data into the Product Dimension of the Staging Area;
- Apply the **Aggregate** Transformation to find out the average costs and the minimum/maximum bottle volume for each sub-category. The output will be loaded into a new Flat File, which is always updated after each run.

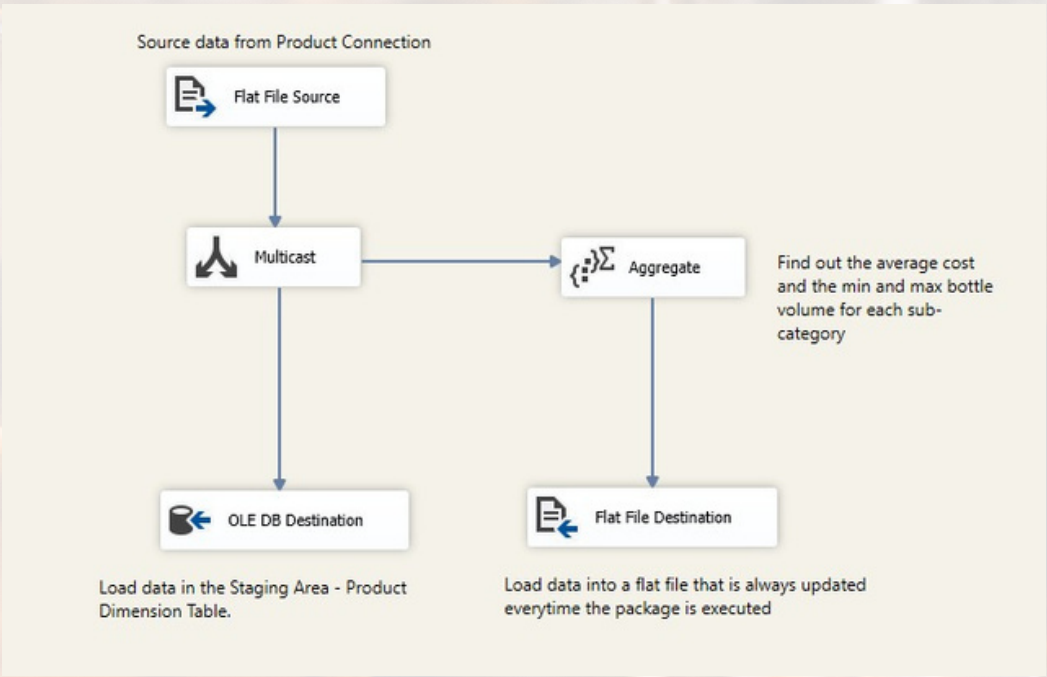


Figure 11 - Loading of Product Dimension.

- **Load Dim Store**

The Store Dimension requires data from two different Flat File sources - both files have the same columns, however, the records were separated.
Since the goal is to combine all the rows into a single output, and no reordering of rows is necessary (since they are already in the required order), the most adequate transformation is the **Union All**.
Afterward, the combined data is loaded into the Store Dimension in the Staging Area.

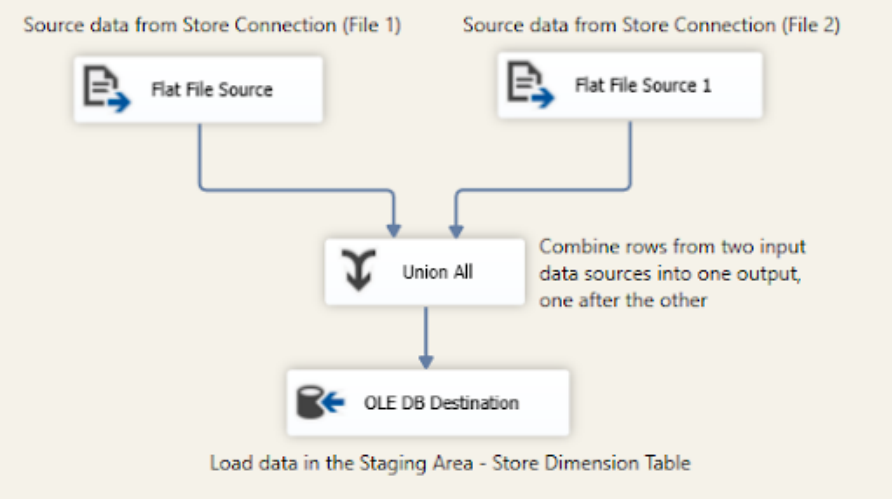


Figure 12 - Loading of Store Dimension.

- **Load Dim Vendor**

In order to have all the data necessary for the loading of the Vendor Dimension, it was necessary to merge data from two different sources:

- Source data from Vendor Connection - Vendor ID and Vendor Name.
- Source data from Vendor Connection (Manager Info) - Vendor ID, Manager Name, and Contact Details

The transformation required in this case is the **Merge Join**, which joins data from two inputs into one output. Firstly, the input data of both tables are sorted in ascending order according to the Vendor ID.

After the transformation, the data is loaded in the Staging Area - Vendor Dimension Table.

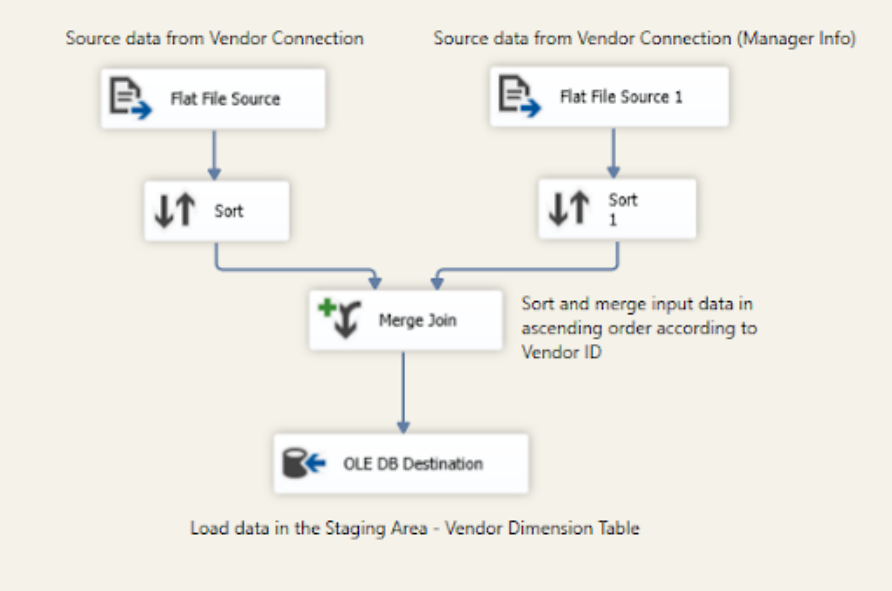


Figure 13 - Loading of Vendor Dimension.

- **Load Dim Date**

Firstly, the data source is extracted from the respective Flat File.
The date source contained values from 2017-2021. However, since the transactions on this project only range from 2018-2021, the data was **filtered** to only present that range of years.
Finally, the transformed data was loaded in the Date Dimension of the Staging Area.

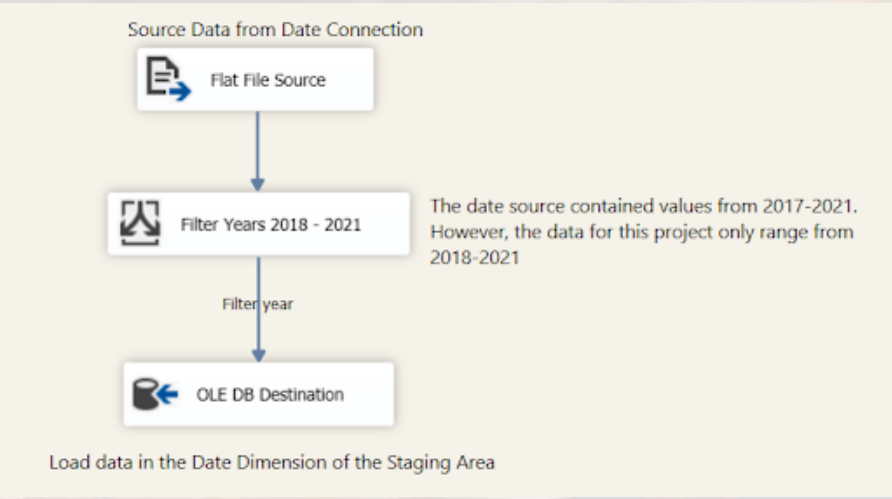


Figure 14 - Loading of Date Dimension.

Order	Output Name	Condition
1	Filter year	(Year > 2017) && (Year < 2022)

- **Load Dim Shipping**

The Shipping Dimension also requires data from two different sources in order to have all the necessary information for the loading process:

- Source data from Shipping Connection - Shipping ID, Shipping Type, and Shipping Class
- Source data from Shipping Connection (Carrier Info) - Shipping ID and Carrier Name

Once again, it was necessary to use the transformation **Merge Join,** so that we can have an output with all the required information. As explained before, the data first needs to be sorted to then be merged. In this case, the input data was sorted and merged in ascending order according to the Shipping ID. Lastly, the transformed data is loaded into the Shipping Dimension Table of the Staging Area.

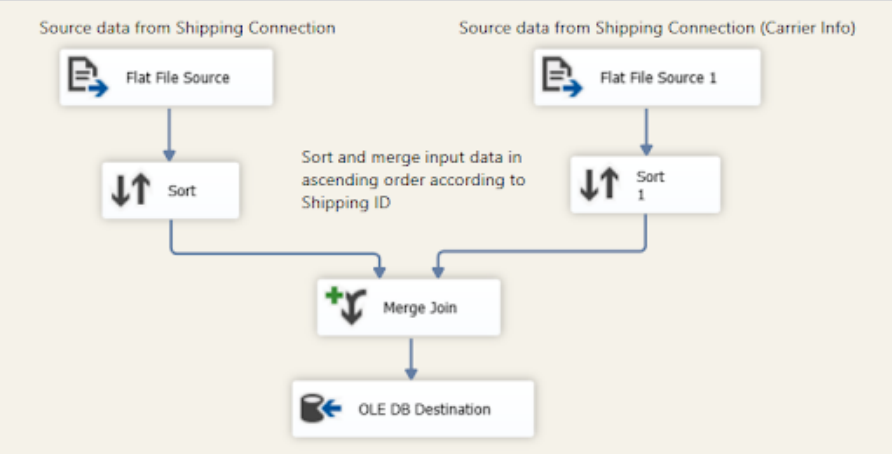


Figure 15 - Loading of Shipping Dimension.

Input	Input Column	Output Alias
Sort	Shipping ID	Shipping ID
Sort	Shipping Type	Shipping Type
Sort	Shipping Class	Shipping Class
Sort 1	Carrier Name	Carrier Name

- **Load Fact Sales**

The information to load the Fact Sales was mostly on the Flat File **Purchase Orders**. However, some of the measures defined previously required inputs from the Flat File **Product** to be calculated - necessary to get State Bottle Cost from the Product Connection, to be able to calculate the cost amount and consequently profit amount.

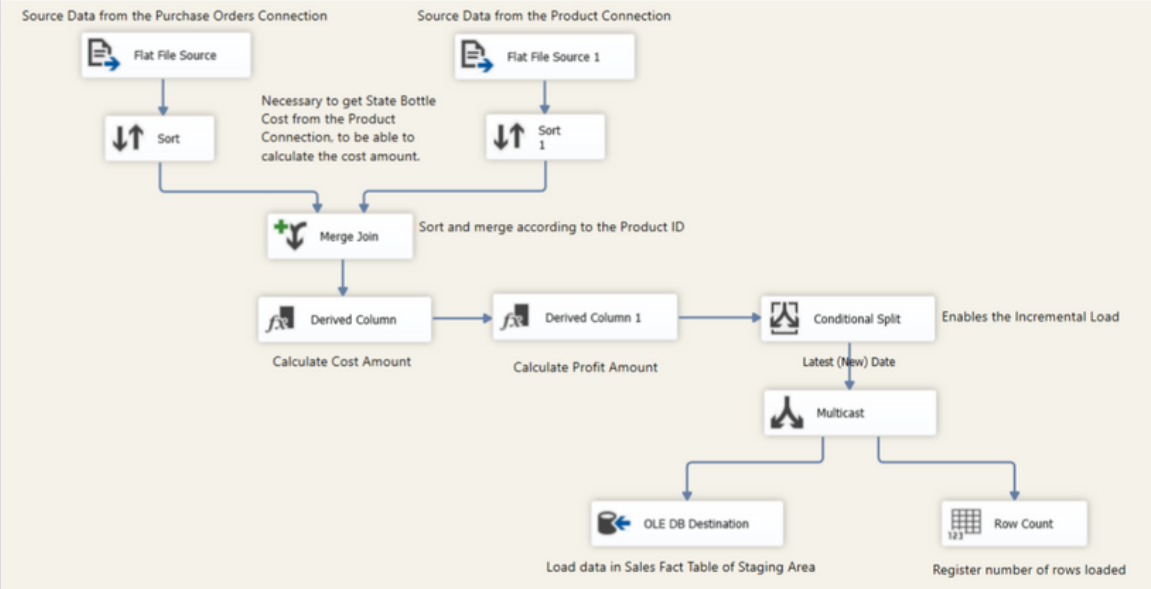


Figure 16 - Loading of Sales Fact Table.

This explains the need for the **Merge Join** Transformation once again. Firstly, the input data was sorted and merged according to the Product ID.

Next, the **Derived Column** transformation was applied with the goal of creating a new column called Cost Amount. To do that, the expression below was used:

Derived Column Name	Derived Column	Expression	Data Type	Length	Precision	Scale	Code Page
Cost Amount	<add as new column>	[State Bottle Cost] * [Bottles Sold]	numeric [DT_NUMERIC]		38	0	

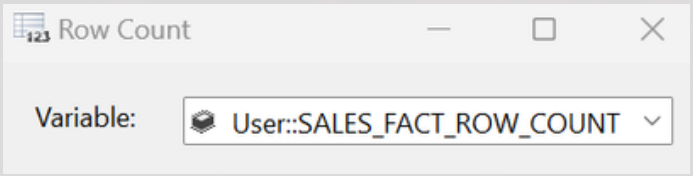
Afterward, another **Derived Column** transformation was created, now with the goal of creating a new column called Profit Amount, which is calculated with the following expression:

Derived Column Name	Derived Column	Expression	Data Type	Length	Precision	Scale	Code Page
Profit_Amount	<add as new column>	[Sale (Dollars)] - [Cost Amount]	numeric [DT_NUMERIC]		38	0	

The next transformation was the **Conditional Split**, which will enable the Incremental Load. The condition below was created so that the system checks the date of the last record loaded and only uploads records that still haven't been loaded into the data warehouse, in other words, only uploads the most recently created or updated data.

Order	Output Name	Condition
1	Latest (New) Date	Date > @[User::Latest_Sales_Loaded_Date]

Finally, the **Multicast Transformation** will direct the inputs to two distinct outputs. The first one will load the records in the Sales Fact Table of the Staging Area and the second one will register the number of rows loaded in the variable below:



5.2. DATA WAREHOUSE LOADING

After cleaning and transforming the data in the ETL process, the following step is to **load this data into the Data Warehouse**.

The main point at this stage is based on the addition of the Surrogate Keys to the dimension tables, allowing the creation of relationships between the dimension and the fact tables. This is a very complex process since it involves the transfer of large amounts of data.

5.2.1. DATA WAREHOUSE - CONTROL FLOW

As mentioned before, we started by doing the **Full Load** process and then made the required changes for the Incremental Load. In the Full Load, the process requires the truncation of the Fact Sales and then the deletion of the dimension tables.

Due to the fact that in the DW there are connections between tables, it is not possible to delete facts and dimensions simultaneously, so it is mandatory that we first truncate the facts, and only after the dimensions.
Below we have this process, which was disabled later on since the Incremental Load process doesn't require it and starts on the loading of the tables.

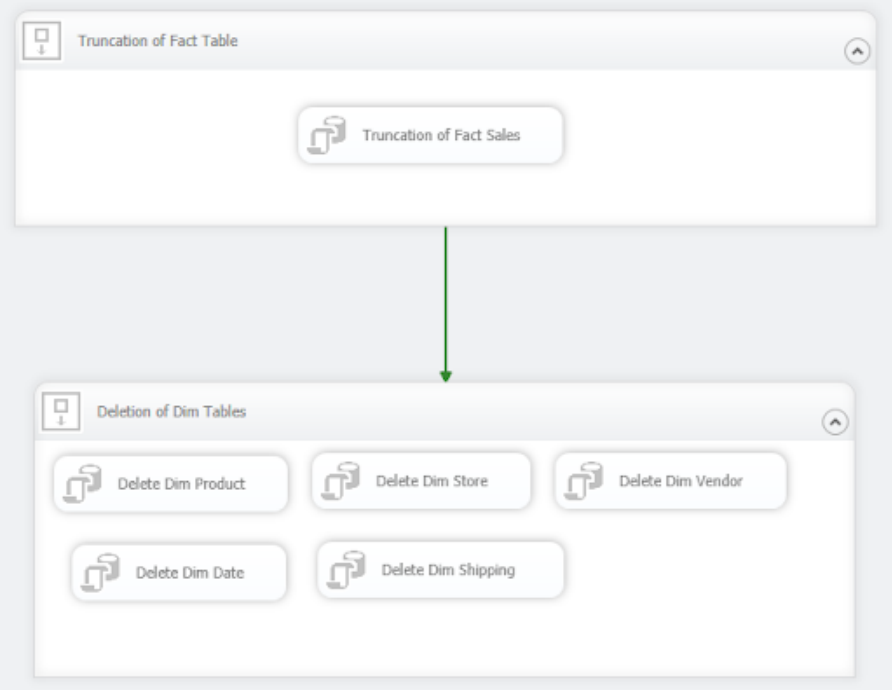


Figure 17 - Deletion and Truncation DW.

As before, the first task in this package is the **Log Start of DW ETL**, which registers the start of the ETL process in the table Log_Stg_Etl.

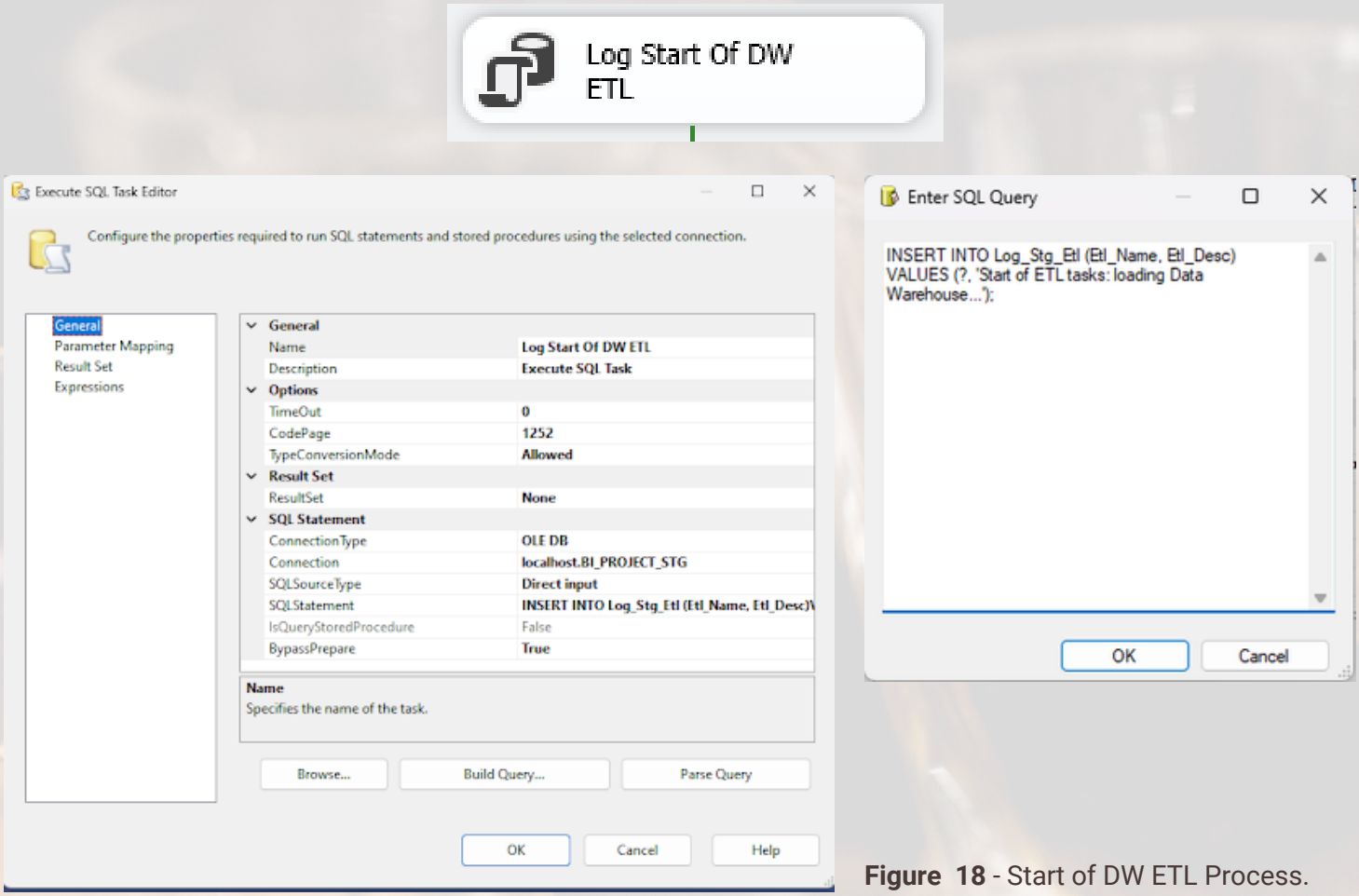


Figure 18 - Start of DW ETL Process.

The process is carried out in a different order from the loading of the Staging Area. At this stage, we can only load the Fact Table after the Dimension Tables are loaded since now the tables have relationships and the Fact Table requires Lookups to retrieve the surrogate keys of the Dimension tables.

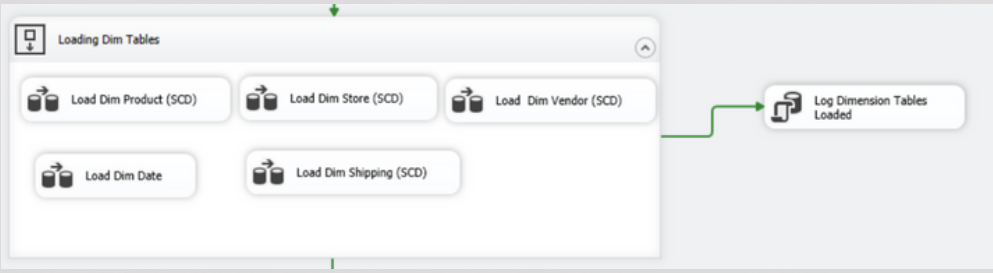


Figure 19 - Loading of Dimension Tables.

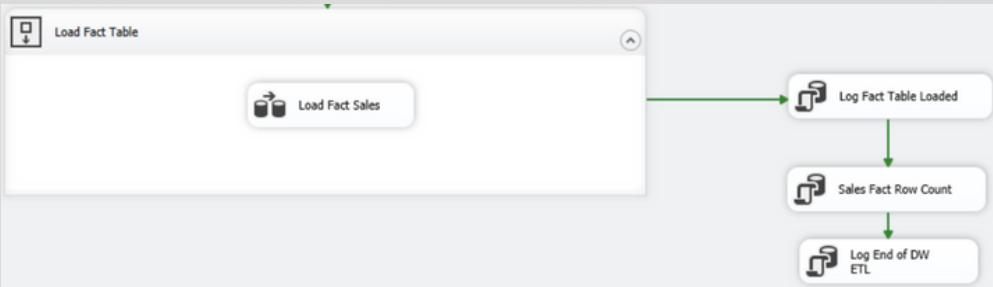


Figure 20 - Loading of Fact Table.

Once again, we created a data flow task to register the loading of the dimension tables - **Log Dimension Tables Loaded** - and another one to register the loading of the Fact Table - **Log Fact Table Loaded.**

Also, in order to register the number of rows loaded for the fact table, we proceeded with the creation of the data flow task **Sales Fact Row Count**.

Finally, the task **Log End of DW ETL** was created to register the end of the DW ETL process.

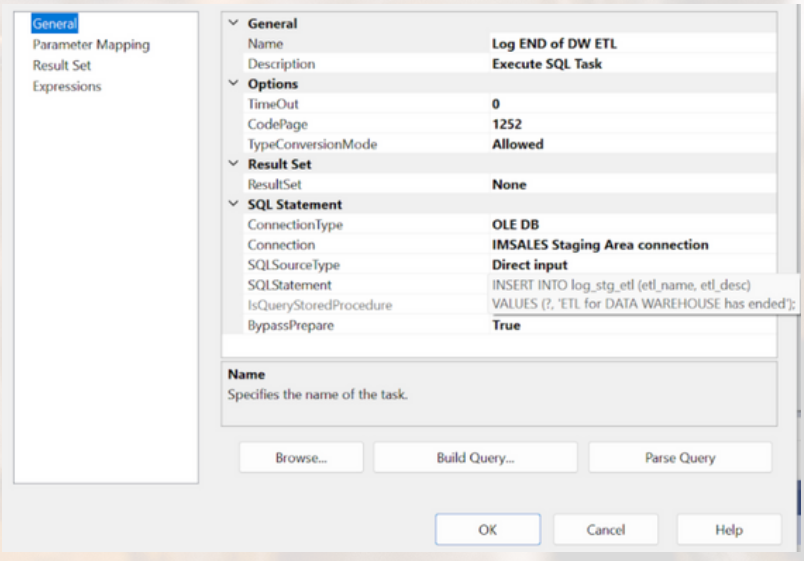
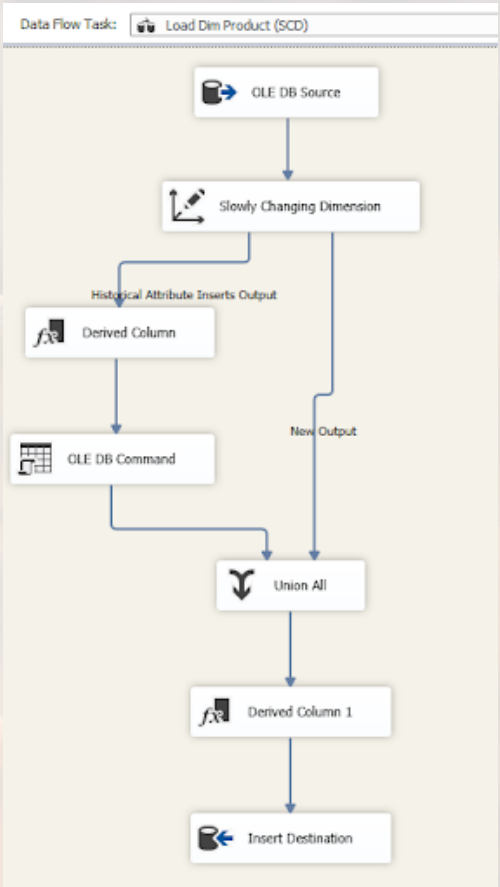


Figure 21 - Configuration of the end of the DW ETL process.

5.2.2. DATA WAREHOUSE - DATA FLOW

The loading for the data warehouse requires particular attention to what type of Slowly Changing Dimensions we are dealing with. As defined in the Slowly Changing Dimensions section, the attributes of the dimensions are either Type 0 (Fixed Attributes) or Type 2 (Historical Attributes).
As mentioned before, for Type 0, the attributes should not change even if new data is uploaded. For Type 2 it was necessary to add two new columns to know which is the most recent and active record: **SCD_Start_Date and SCD_End_Date**. The column SCD_Start_Date will show the date when the attribute value was updated, and SCD_End_Date refers to the last day before it was changed. If a value has not suffered any change, the SCD_End_Date is set to NULL.

Loading of Product Dimension (SCD)
We start by extracting the information from the table Stg_Dim_Product, which is the product dimension of the staging area. In this case, the Slowly Changing Dimension transformation was used, which has the purpose of helping manage change in the values in the columns.



First, define the BK_Product as the business key and then proceed with the configuration of the slowly changing dimension attributes.
In this dimension we have both **Type 0** SCD, which are the fixed attributes, and **Type 2** SCD, known as historical attributes.
The fixed attributes are defined as values that will not change overtime: **Category_Name and Sub_Category_Name.**
The historical attributes, which we want to keep track of the changes are: **Bottle_Volume; Item_Description; Pack; State_Bottle_Cost and State_Bottle_Retail.**
Lastly, the data is loaded in the Dim_Product table of the Data Warehouse.

Figure 22 - Load of Dim Product (SCD).

Loading of Store Dimension (SCD)

We start by extracting the information from the table Stg_Dim_Store, which is the product dimension of the staging area. Afterward, employ the Slowly Changing Dimension.

First, define the BK_Store as the business key and then proceed with the configuration of the slowly changing dimension attributes.

In this dimension, we have only Type 2 SCD, known as historical attributes, which we want to keep track of changes. These are City, County, County_Number, Proper_Address, Store_Name, and Zip_Code.

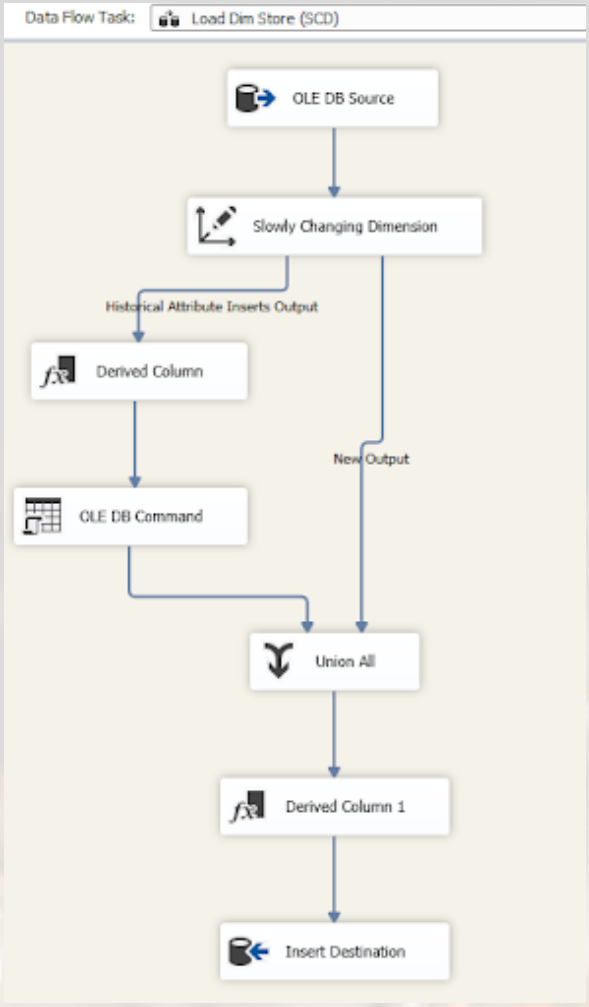


Figure 23 - Load of Dim Store (SCD).

Loading of Vendor Dimension (SCD)

We start by extracting the information from the table Stg_Dim_Vendor, which is the product dimension of the staging area. Connect the Slowly-Changing Dimension to the source and define the BK_Vendor as the business key. Proceed with the configuration of the slowly changing dimension attributes.

In this dimension, we have only **Type 2** SCD, known as historical attributes. These are **Contact_Details, Manager_Name, and Vendor_Name**.

Loading of Date Dimension

The process starts with the extraction of information from the Stg_Dim_Date - date dimension of the staging area. A Slowly Changing Dimension transformation was used.
In this Date Dimension, we also define the business key - BK_Date and move on to the SCD attributes. We opted to only use fixed attributes - Type 0 SCD, as values in columns should not change over time. These attributes are: **Day_Name, Day_Number, Full_Date, Month_Name, Month_Number, Proper_Date, Semester_Name, Semester_Number, Trimester_Name**, **Trimester_Number**, and lastly **Year**.
The data was then loaded into the Dim_Date table of the Data Warehouse.

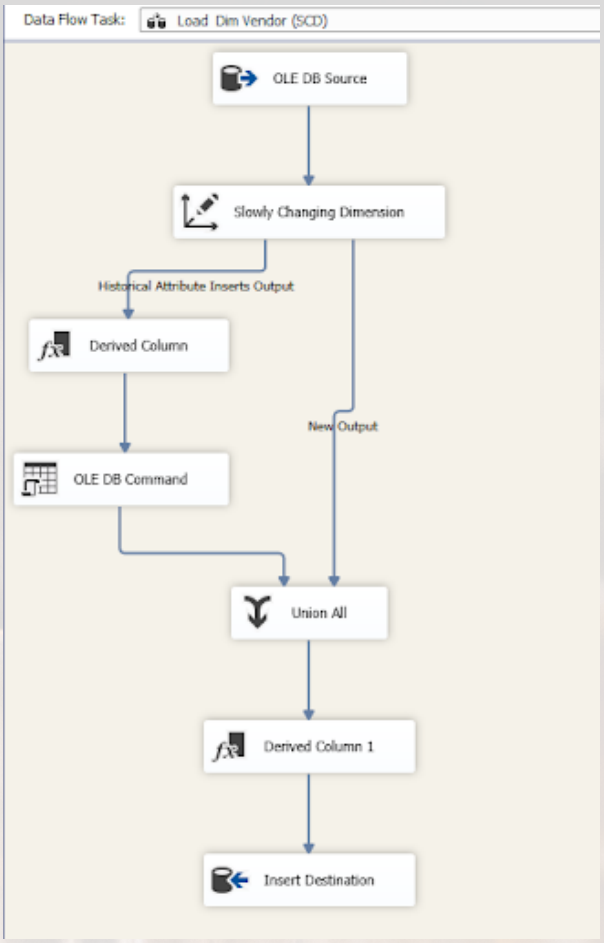


Figure 24 - Load of Dim Vendor (SCD).

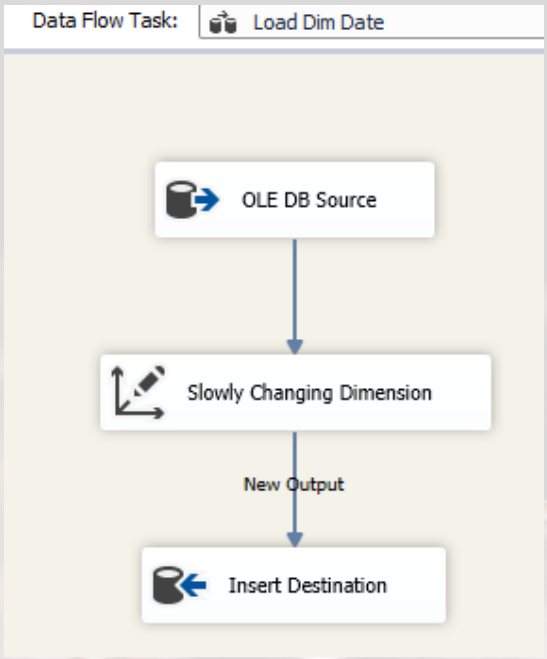


Figure 25 - Load of Dim Date (SCD).

Loading of Shipping Dimension

In this last dimension, the concept is similar, as we start by extracting the information from the Stg_Dim_Shipping table and also use a Slowly Changing Dimension to manage values. The business key is set as BK_Shipping.

The shipping dimension has one historical attribute - Type 2 SCD, which is the **Carrier_Name**, and two Fixed attributes - Type 0, for the **Shipping_Category** and **Shipping_Class**.

After the configuration of the SCD attributes, we pass on the loading of the data into the Dim_Shipping table in the Data Warehouse.

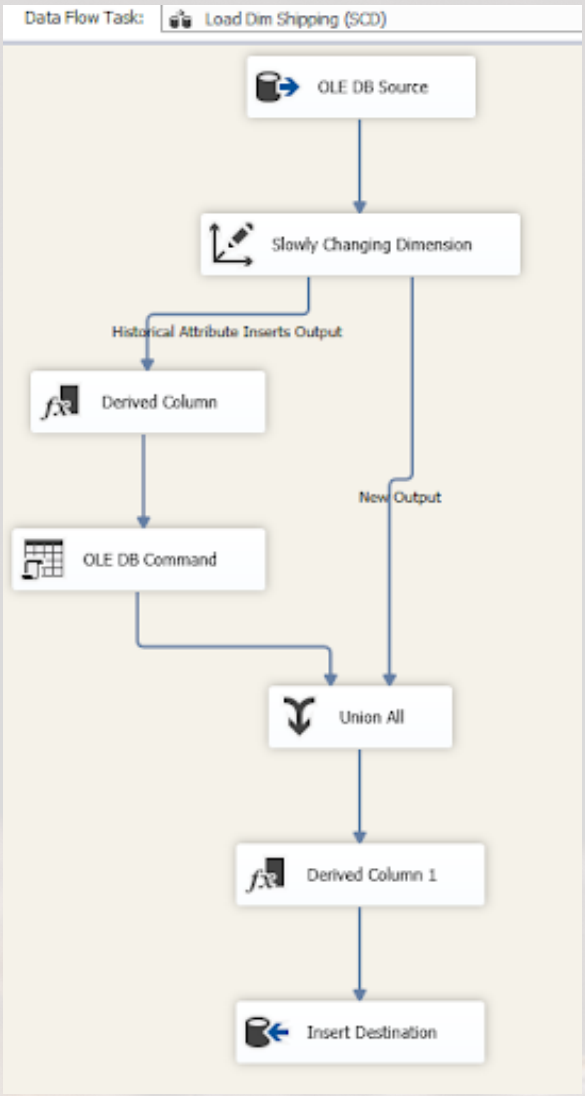


Figure 26 - Load of Dim Shipping (SCD).

Loading of Fact Sales

With all dimensions being loaded, we now pass on to loading the fact table, which in our case is Sales. To do so, a lookup function was used to access the surrogate keys of each of 5 dimensions created in this project. As we can observe in the image below, we performed a lookup task for each key we wanted to access and then loaded the data into the Fact_Sales table in the Data Warehouse.

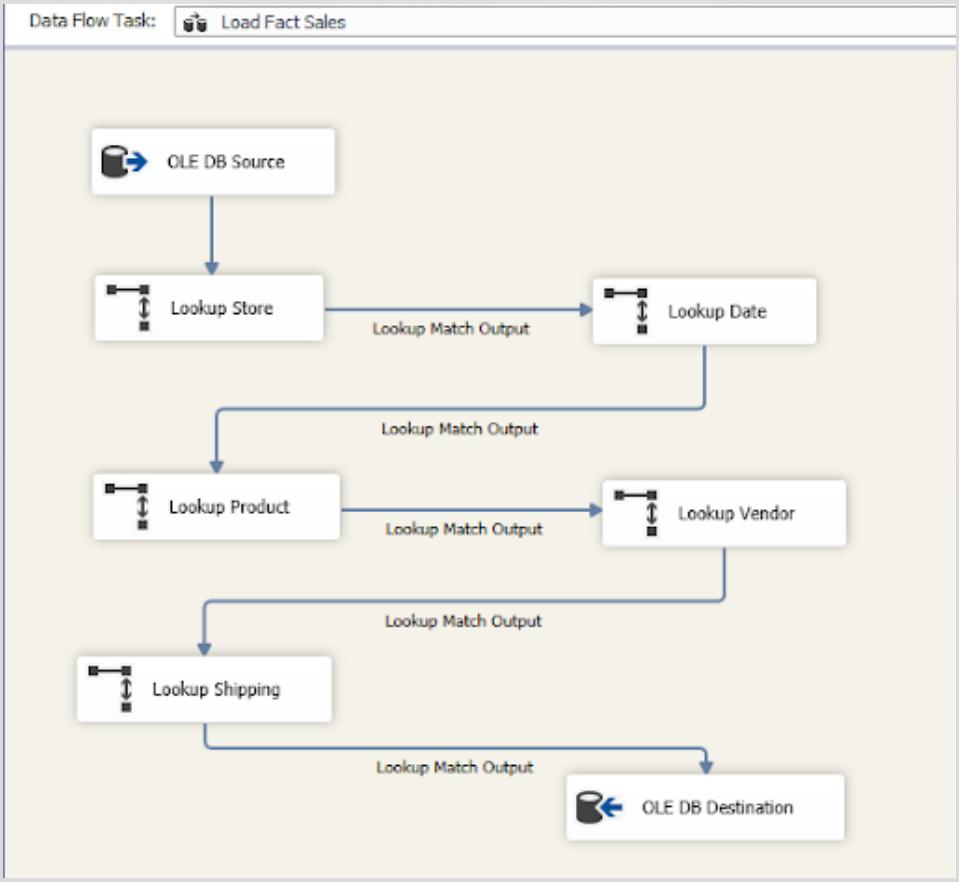
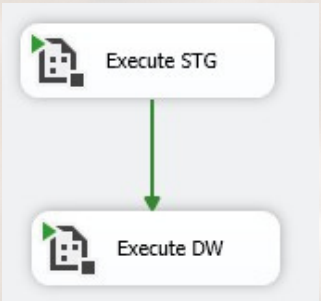


Figure 27 - Load of Fact Sales.

5.3. MASTER PACKAGE

With the goal of executing both packages, the ETL process of the Staging Area first and the ETL process for the Data Warehouse later, we have created a Master Package.



We also employed a **Script Task** that will make a pop-up appear on the screen after each package is executed successfully.

```
ST_b22f872ae36545269534597aac56080b | ST_d4cb163ca9714fec8c55c148a65666a1.ScriptMain | Main()

Help:  Introduction to the script task

Namespaces

namespace ST_d4cb163ca9714fec8c55c148a65666a1
{
    /// <summary>
    /// ScriptMain is the entry point class of the script.  Do not change the name, attributes,
    /// or parent of this class.
    /// </summary>
    [Microsoft.SqlServer.Dts.Tasks.ScriptTask.SSISScriptTaskEntryPointAttribute]
    0 references
    public partial class ScriptMain : Microsoft.SqlServer.Dts.Tasks.ScriptTask.VSTARTScriptObjectModelBase
    {
        Help:  Using Integration Services variables and parameters in a script

        Help:  Firing Integration Services events from a script

        Help:  Using Integration Services connection managers in a script


        /// <summary>
        /// This method is called when this script task executes in the control flow.
        /// Before returning from this method, set the value of Dts.TaskResult to indicate success or failure.
        /// To open Help, press F1.
        /// </summary>
        0 references
        public void Main()
        {
            // TODO: Add your code here
            MessageBox.Show("DW Executed");
            Dts.TaskResult = (int)ScriptResults.Success;
```

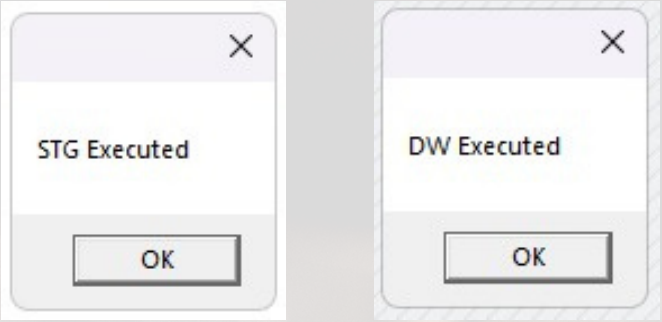


Figure 28 - Script Task configuration.

5.4. EXECUTED PACKAGES

The steps necessary for creating the Staging Area and Data Warehouse are, at this point, completed. The figures below show the final diagrams, executed and with no errors.

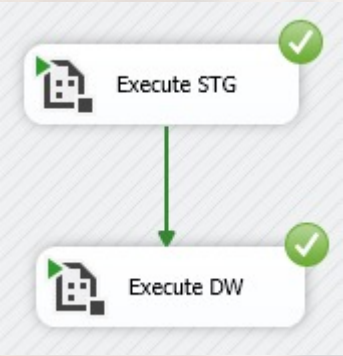


Figure 29 - Executed Master Package (Staging Area and Data Warehouse).

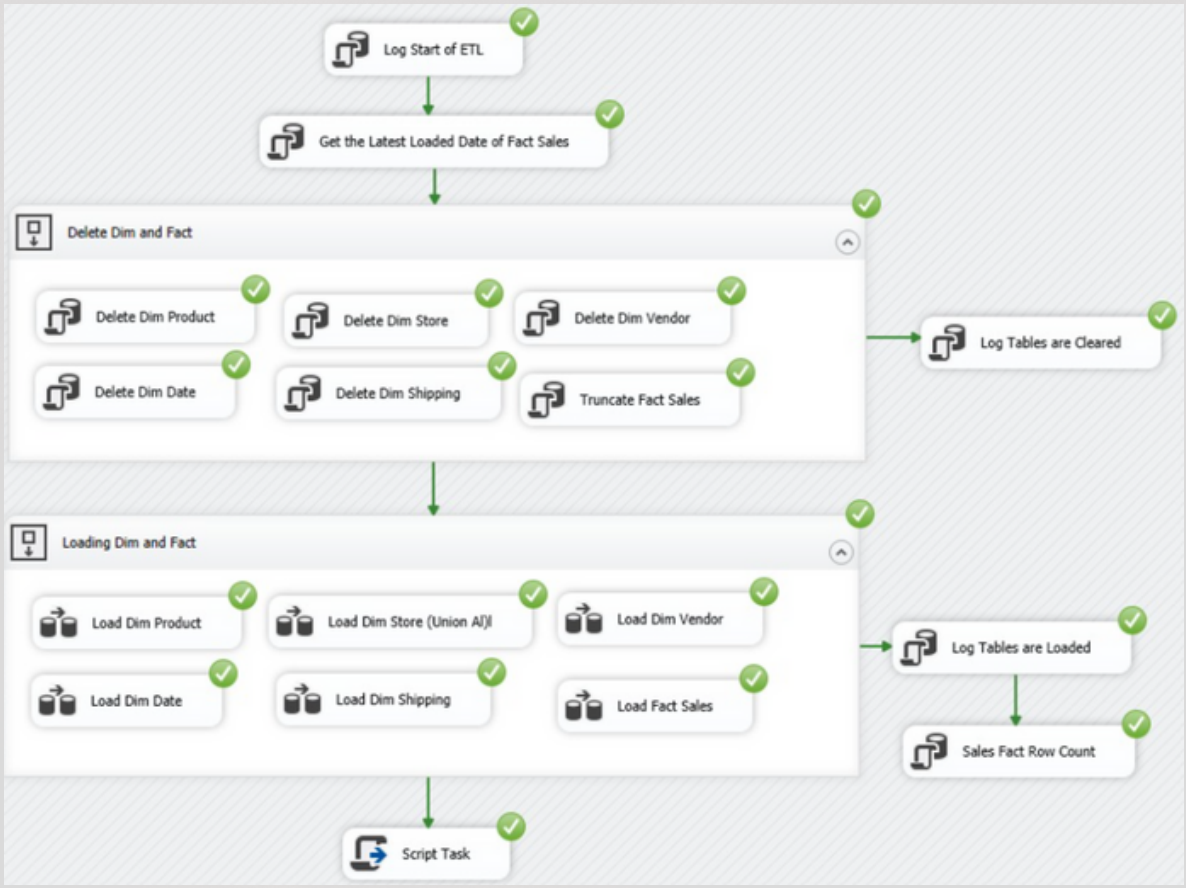


Figure 30 - Executed Staging Area.

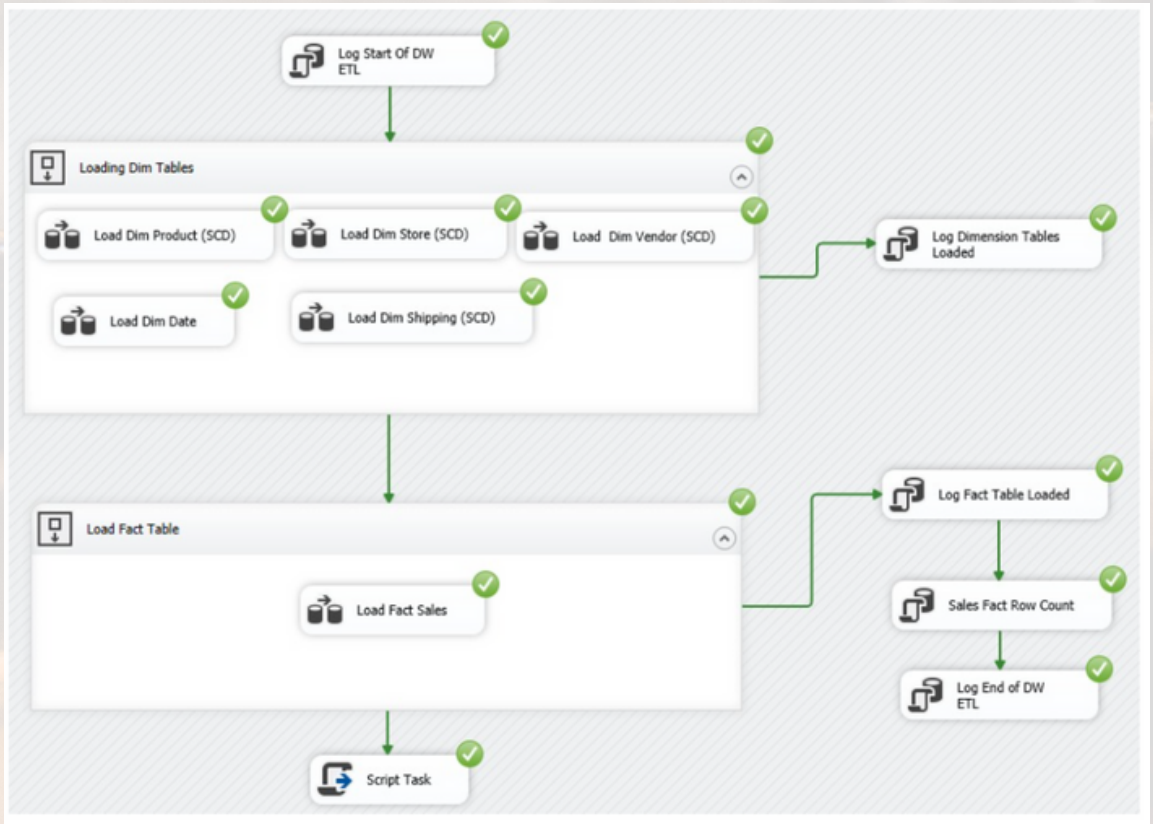


Figure 31 - Executed Data Warehouse.

06.
CRITICAL REVIEW
LESSONS LEARNED

1. Initially, we had an excel file with several sheets. When creating the connections we had errors, so we opted to split each sheet into a CSV file.
2. As our database was already very well organized, in order to be able to apply the various transformations, we split some of the CSV's to be able to apply the transformations merge join and union all, for example.
3. We had to make several changes to the scripts throughout the work because some data types were wrong and others didn't match (for example, the proper data when we had to do the lookup was not matching).
4. Initially, we were having trouble running the processes but then, through research, we saw that it was necessary for certain connections to set the Code Page and increase the output column width.
5. Finally, our main problem was that we lost our project midway since the project was not being saved correctly. Fortunately, we were able to recover an old version.

With the accomplishment of this project, we learned the importance of the details identified previously and the importance of saving the project as a whole in the temp file so that it is possible to share the project and that the solution opens with all the correct configurations. We also saw the importance and impact of self-learning - researching and looking at tutorials, forums, and articles was a fundamental process for the success of the project.

06.
CRITICAL REVIEW
CONCLUSION

Firstly, by analyzing the company's business context, needs, and possible challenges, we were able to design a Data Warehouse where we got the opportunity to fully understand dimension modeling and organize it according to the business needs and the necessary requirements for our project.

In the implementation side of things, although there were some demanding tasks with multiple solutions, we consider having made the right decisions in the ETL process, a step that is essential is the construction of the DW. This process involves tools that are able to collect, organize and load high levels data from different sources and across different platforms, something that is absolutely mandatory for data-driven decisions that enable companies to grow in a smart and sustainable way.

Also, the fact that data comes in all sorts of formats and quantities paired with the importance of not making major mistakes and late changes to the data warehouse, showed us the importance of creating a staging area that stores data before loading it to the main DW.

In conjunction with the first part of the project, where we understood what our company of choice needed, the ETL process allowed us to really put theory into practice and create an organized, manageable, and accessible structure that allows for more efficient decisions for future data analysis.